

# **DOE2000 Panel Parallel Programming Tools**

**Rod Oldehoeft  
Advanced Computing Lab  
Los Alamos National Laboratory**

**American Nuclear Society  
Winter Meeting  
November 16, 1998**

# Projects Overview: Tools for Algorithm Development

- Global Arrays: library for manipulating arrays distributed across memories of multicomputers
- Overture: C++ class library for writing PDE solvers using overlapping/adaptive block-structured grids
- POOMA: C++ framework for high-performance, parallel computational science applications

# Projects Overview: Runtime & Other Support Tools

- Cumulvs: facility for adding fault tolerance, steering, visualization to existing programs
- Nexus/Globus: general-purpose support for distributed large scale applications
- PAWS: facility for inter-application parallel data transfer in hetero distributed systems

# Projects Overview: Runtime & Other Support Tools

- SILOON: aid for using scripting languages in accessing libraries and frameworks
- TAU: support for instrumenting, tracing and analyzing parallel programs
- Tulip: runtime layer with threads and data transfer facilities

# Projects Overview: Developers' Tools

- ATLAS/PHiPAC: automatic generation of optimized Basic Linear Algebra routines
- PADRE: interface to several data distribution methods for parallel multicomputers
- PETE: extensible implementation of C++ expression template technique

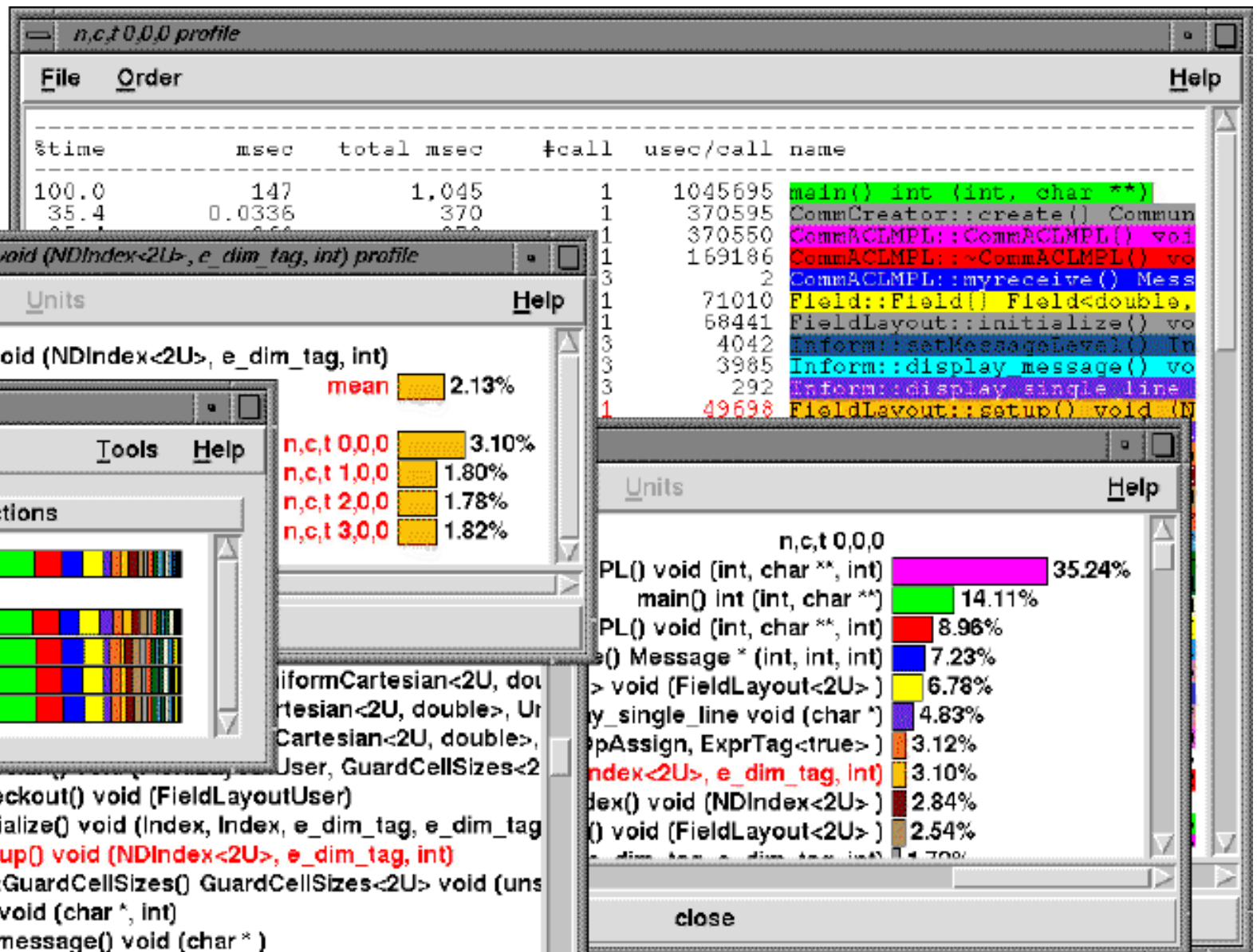
# Selected Tool: POOMA

- C++ framework provides abstractions: fields, grids, particles as well as arrays
- Array implementation defined by plug-in "engines": F90 and various distributions
- App writer writes code invariant to them
- C++ facilities specialize during compilation for efficient data access
- Used in several computational physics applications



# Selected Tool: TAU

- Environment for performance tuning
  - Instrument source code
  - Run program
  - Use visualization tools to analyze performance
- Variety of measures available
  - Time spent in each subprogram
  - Other hardware counters instead of clock
  - User-defined events
- Several TAU viewers, or use Vampir
- C, C++ now, Fortran90 imminent



# Selected Tool: SMARTS

- Task + data parallel execution on SMPs
- Subset Pthreads, HPC++ threads APIs
- Iterates for data or loop-level parallelism
  - Very lightweight, run to completion
  - Dataflow graph gives Iterate execution order
  - Optimizes execution to improve cache hits
- Usable directly: SPPM uses Pthreads API
- Usable by other software: POOMA II



# Selected Tool: PAWS

- Coupling separate parallel applications
  - Running on same or different machines
  - Running on heterogeneous machines (Nexus)
  - Different parallel data distributions
- Parallel data transfer with simultaneous redistribution
- Minimal change to components
- PAWS controller coordinates interactions
- Integrated with POOMA, visualization libs

PAWS

```

*****
The PAWS Controller, version 1.0
For more info: http://www.acl.lanl.gov/paws/
*****
Running femm ...
Running fifea ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...
Waiting for femm and fifea to register with controller ...

```

FEMM

```

Receiving paws_tmat ...
In femFrameFunction gonna updatedata
In femGeomUpdate buff 1 tmat is
-0.839071 0.544021 0.000000 0.000000
-0.544021 -0.839071 0.000000 0.000000
0.000000 0.000000 1.000000 0.000000
0.000000 0.000000 0.025000 1.000000
-0.839071 -0.544021 0.000000 0.000000
0.544021 -0.839071 0.000000 0.000000
0.000000 0.000000 1.000000 0.000000
0.010000 0.010000 0.000000 1.000000
In femCalcColor set buff 1
done femSwapdatabuff back 0 front 1
Receiving paws_zx ...

```

FIFEA

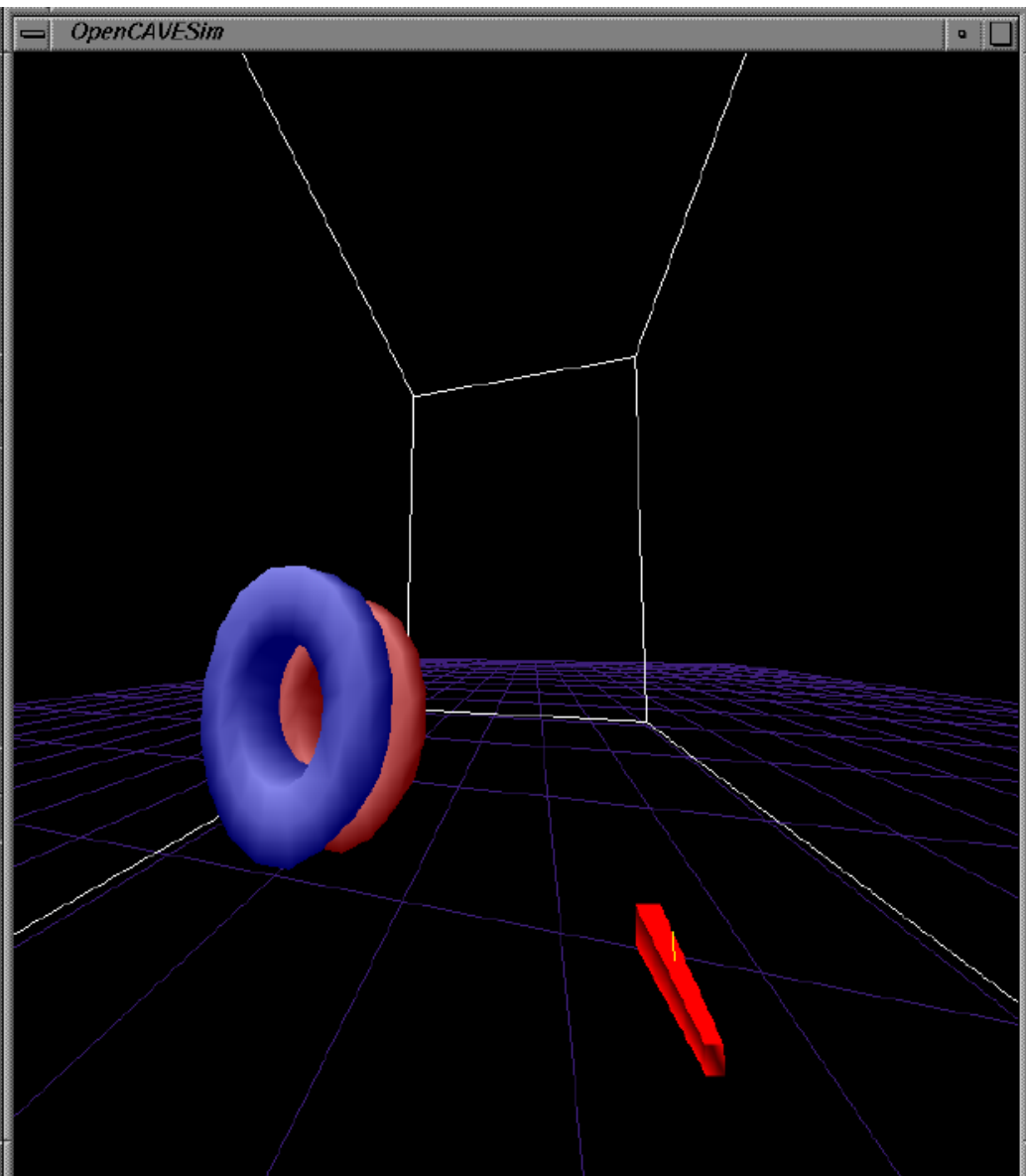
```

fifea: sending data via fifea connection at step 500
PAWS(00) Sending dynamic data via PAWS ...
PAWS(00) Sending zx ...
PAWS(00) Sending tmat ...

0 t1=0.499000 dt_in=0.000500 dt_out=0.001000 step=499
dt= 5.0000002374872565E-04 dtp= 530047699546723.4 dtp1=
3.4028234699999998E+38

0 Pigeon Holes restructuring!
0 Bins: 0 0.130007 5 1 0.130007 25 2 0.045900 50
dt= 5.0000002374872565E-04 dtp= 530047699546723.4 dtp1=
530047699546723.4
dt= 5.0000002374872565E-04 dtp= 530047699546723.4 dtp1=
530047699546723.4
dt= 5.0000002374872565E-04 dtp= 530047699546723.4 dtp1=

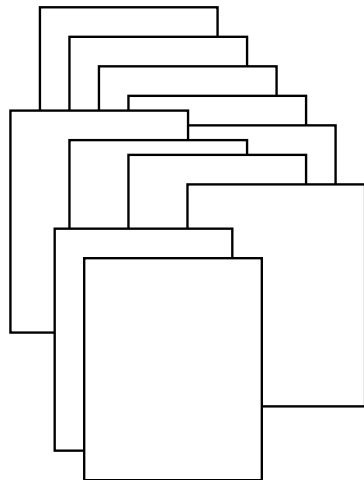
```



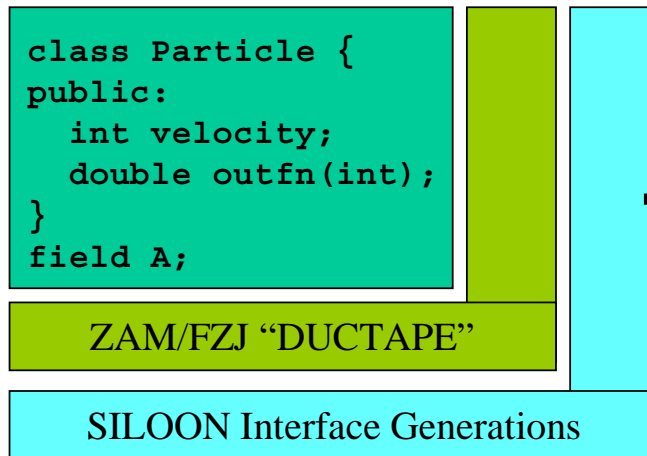
# Selected Tool: SILOON

- Rapid prototyping by making libraries accessible from scripting languages: Python, Perl, TCL, etc
- E.g., this C++ code:  
*int foo( double d) { ...; return(x); }*  
is accessible from Perl:  
*\$aDouble = 4.5; \$error = 0;*  
*\$intR = siloon::invoke("int foo(double)",*  
*\$aDouble, \$error);*

**Parallel C++ Class Library**  
(POOMA/OVERTURE/PAWS)



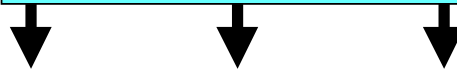
**External Interface**



```
class Particle {  
public:  
    int velocity;  
    double outfn(int);  
}  
field A;
```

ZAM/FZJ "DUCTAPE"

SILOON Interface Generations



**glue.pl**

Perl  
run-time  
glue

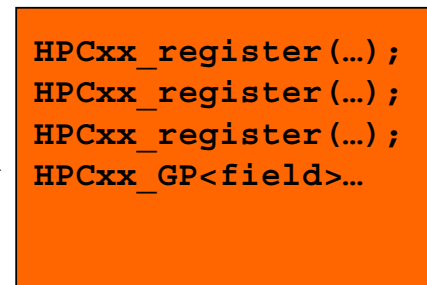
**glue.tcl**

TCL  
run-time  
glue

**glue.py**

Python  
run-time  
glue

**Generated C++ RTS Glue**



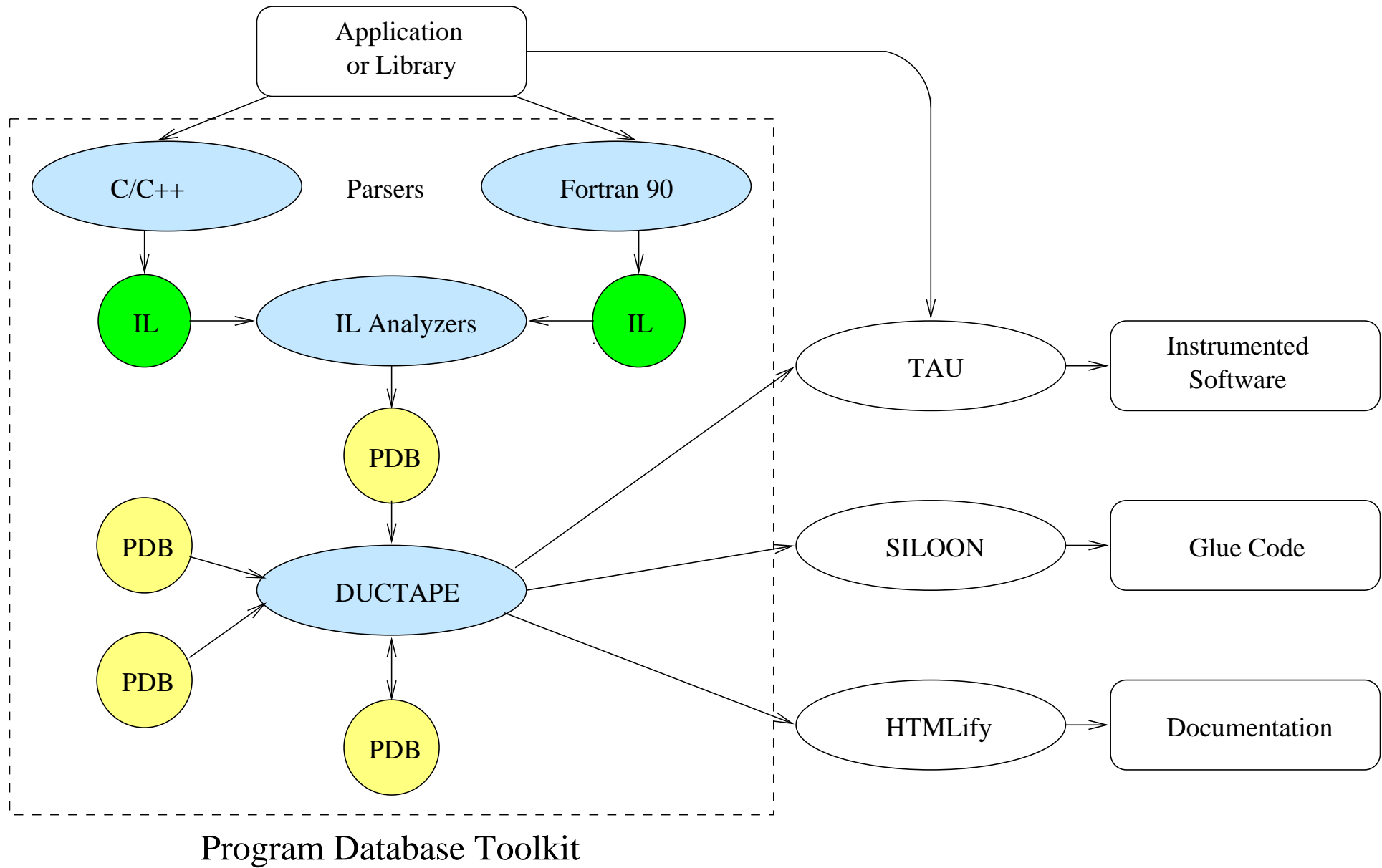
compile



executable

# New Approach: Program Database Toolkit

- Tools know too little about the programs they're applied to
- PDT extracts info for use by other tools
  - Parsers for F90, C, C++ generate IL
  - IL analyzers extract needed info to PDB files
  - DUCTAPE provides API for tools to get info
- Current tools rethought, new tools coming
- This is a powerful idea!



# **New Approach: Component Architectures for HPC (DOE2000 CCA Forum)**

- Want advantages of commercial "application builders" in HPC world
- Cannot tolerate their inefficiencies
- InDEPS and Ligature, two early designs:
  - Component description syntax describes parts
  - Combining infrastructure handles parts
  - Bindings are "glue code" for parts' I/O
  - Repository stores info, including performance
  - Building mechanism is user interface
- This is a powerful idea!

